Development Team Project: Design Document

Secure Software Development

MSc Cyber Security

Team Availability

Jonathan Ashmore

Osarodion Samuel Tolofari

Abidemi Adelakun

University of Essex

29 August 2022

Introduction

In modern times, paper-based logbooks have been replaced by digital versions such as Navtor for marine ships (NAVTOR, 2022). In addition, the International Space Station (ISS) has complex systems for recording telemetric events, conditions, and actions; however, other types of logbooks are kept for personal and scientific space-related research, such as the Behavioural Issues Associated with Isolation and Confinement (Nasa.gov, 2011), and the European Space Agency's Samantha Cristoforetti's personal logbook (outpost42.esa.int, 2015).

Overview

This project aims to design a secure application that helps ESA astronauts on the ISS record sensitive research information in line with OWASP's main security threats (OWASP, 2021). In addition, it will help researchers and selected ESA personnel on Earth have remote access to the logbook for downloading information.

Approach and Methodology

We propose an agile approach using the secure scrum methodology to achieve agility and security in web application development using the OWASP top 10 security risks as a benchmark for web application security (See Figure 1). This approach ensures that the software is developed using iterative and incremental development processes with a special focus on securing the software through the entire software development process as seen in Figure 2 (Pohl & Hof, 2015).

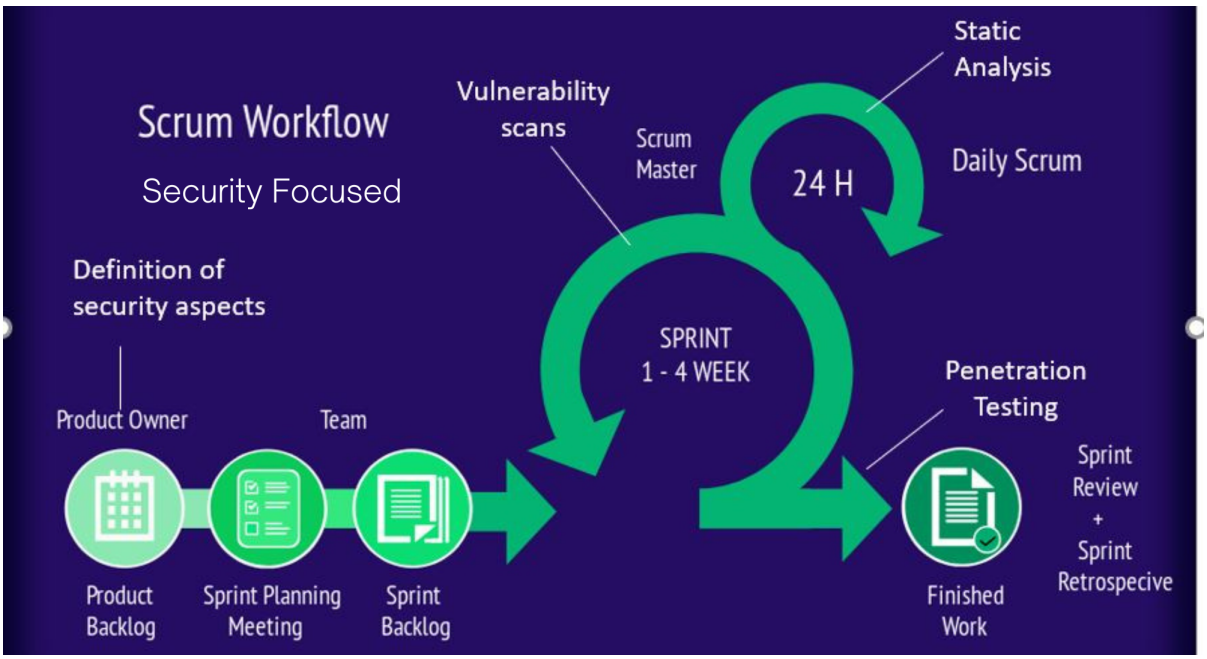


Figure 1: Security Focused Scrum Workflow (SoftScheck, 2022).

Development Process

Product Backlog	Sprint Planning	Implementation	Sprint Review & testing	Sprint Retrospective
Product Backlog	Threat Modelling and Risk Assessment	Daily Standup Meetings	Sprint review meetings	Sprint retrospective meeting
User stories description and prioritization (functional user stories and Security related user stories)	Tasks breakdown and assignment	Static and dynamic code Analysis	Penetration Testing	Review Documentation
System Requirements	Sprint backlog	Pentesting and User Testing	Code Review	Technical Security documentation and test reports
Security and Privacy Requirement Analysis	Sprint goal	Identify and mitigate risks	Product demonstration	Definiation of done
Risk Analysis	Document assigned tasks, owners and due dates		Testing	Potentially Shippable Product Increment

Figure 2. Development Process (Maier, 2017).

Risk Assessment

The risk assessment for the development of the application has been evaluated, and each factor has been considered based on its likelihood and impact; the entire assessment is available in Appendix G.

Hardware

The ISS has a high-technology IT infrastructure, including next-generation microprocessors, interface buses, data networks, and SpaceWire: a 200Mbps high-speed data link (esa. int, 2022). Furthermore, ESA and ISS astronauts are equipped with high-end IBM Thinkpads and HP Zbooks using Linux and Windows operating systems (Space Exploration Stack Exchange, 2014). We have opted to use the cloud platform as a service infrastructure to host this application to achieve the following benefits (See Appendix I).

Assumed vulnerabilities and mitigation steps

In designing this application, secure design patterns are planned to utilise threat modelling (STRIDE) and reference architectures to prevent security gaps.

With the OWASP top 10 vulnerabilities (2021) alongside the threat modelling below, the mitigating steps are to be taken, as shown in Figure 3 (threat modelling) and Table 1 (OWASP vulnerability).

Threat Modeling of the risk in the application

	Threat	Property violated	Threat Definition
S	Spoofing	Authentication	impersonating someone
T	Tampering	Integrity	Modification of the application
R	Repudiation	Non -repuiation	Not accepting responsibility
I	Information disclosure	Confidentiality	Providing information to someone not authorised to access it
D	Denial of service	Availability	Exhausting resources needed to provide service
E	Elevation of privilege	Authorization	Allowing someone to do something they are not authorised to do

(Hewko, 2021)

Figure 3. Application Threat Modelling Risk.

Vulnerability	**Description**	**Mitigative Step Taken**
A01:2021 Broken Access Control	Gaining access to user accounts and impersonating users and administrators, and that regular user can gain unintended privileged functions (Learning Center, 2022) Spoofing threat.	• Access is denied by default • Sensitive data will not be stored at the root. • Access control mechanism is employed
A02:2021 Cryptographic Failures	Exposure of sensitive application data on a weak or non-existent cryptographic application (Sengupta, 2022). Information disclosure Threat	• All sensitive data at rest is encrypted using the Werkzeug Security library: Sha256 hashing and salting • Data in transit is encrypted using secure protocols like TLS.

A03:2021 Injection	A web application vulnerability allows unsolicited data, causing that data to be compiled and executed on the server. Tampering Threat	• Server-side input validation will be implemented. • We will implement LIMIT and other SQL controls within queries to prevent mass disclosure of records in case of SQL injection. (Kaplan-Moss and Holovaty, 2008)
A04:2021 Insecure Design	Missing or ineffective security controls. Elevation of privilege Threat	• An established Secure Development Lifecycle is adopted (scrum model). • Automated security tests will be implemented • System and Network Layer separations are implemented.
A05:2021 Security Misconfiguration	Lack of security hardening across the application. Denial of Service Threat	• Minimal setups without unnecessary features and components (Kaplan-Moss and Holovaty, 2008).
A06:2021 Vulnerable and Outdated Components	When a software component is unsupported, out of date, or vulnerable to a known exploit	• Unnecessary features, components, files, and documentation were minimised. • Dependency checkers were used to check

		outdated/vulnerable components (F5.com, 2022)
A07:2021 Identification and Authentication Failures	Functions related to a user's identity, authentication, or session management are not implemented Correctly or not adequately protected by an application.	• Multi-Factor Authentication (MFA) • CAPTCHA challenge • Enhanced authentication and session management are adopted.
A08:2021 Software and Data Integrity Failures	Using codes and infrastructure that do not protect against integrity violations	• content profile validation, such as the XML and JSON content profiles, is configured. • NGINX software components were installed from trusted sources.
A09:2021 Security Logging and Monitoring Failures	Failure to sufficiently log, monitor, or report security events, such as login attempts	• Application requests are logged • Remote logging will be implemented.

A10:2021 Server-Side Request Forgery	The web application is fetching a remote resource without validating the user-supplied URL	• Attack Signatures will be assigned to Security Policies • ACLs will be used to enforce user restrictions to host and port combinations by configuring access control entries (ACEs). • VPN will be used to access the app remotely.

Table 1. OWASP Top 10 vulnerabilities and mitigations. OWASP (2021).

Applicable framework

This application will be designed using EU GDPR Compliance Criteria adopting the ISO 27002 and the NIST 800-53, incorporating the principles relating to Personal Data, Processing activities and Processing Security (secure-controls framework, 2018). In addition, we have proposed the ISO 29100 and the General Accepted Privacy Principle for the lawfulness of processing data and the right to portability as shown in Figure 4 below.

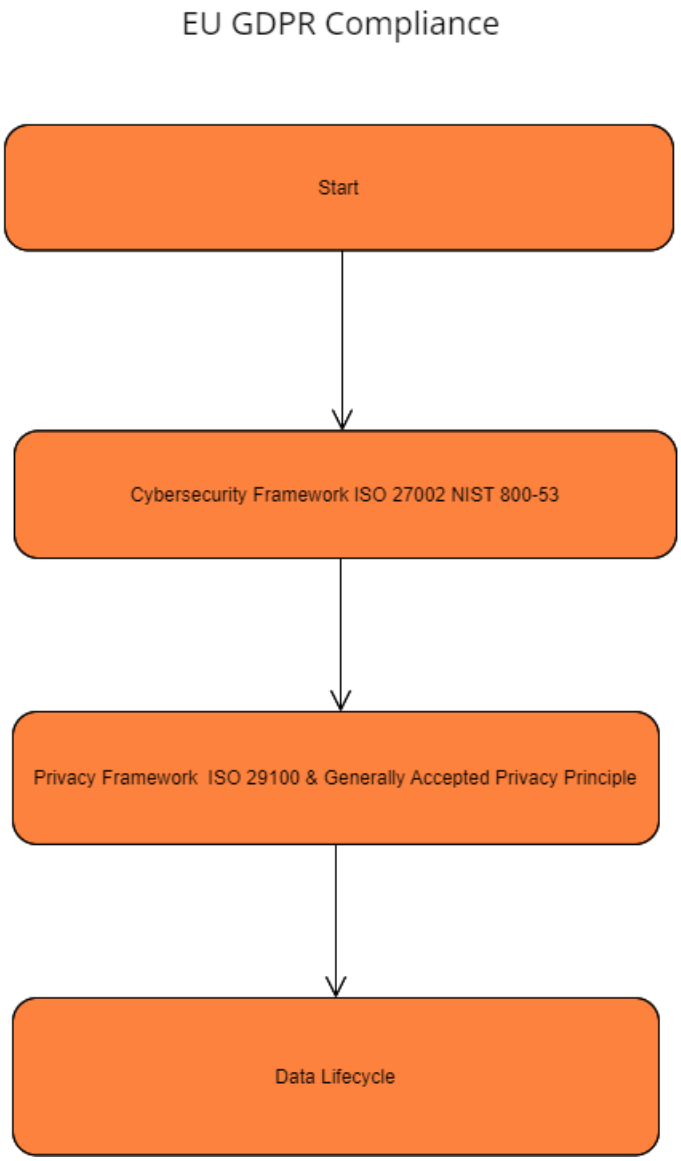


Figure 4 GDPR Compliance. Securecontrolsframework (2018)

Tools and libraries

Python programming language is used due to its open-source focus, readability, libraries, and popularity among novice coders and cyber security personnel (Python.org, 2019). The application will be coded using Visual Studio Code and shared on GitHub and will include the following tools as seen in Table 2.

Choice Of Tool	**Reason For Tool**
Python	An easy-to-use programming language.
Flask	Lightweight web development framework.
Werkzeug Security	Utility for password hashing and salting.
SQLite3	Lightweight serverless embedded database.
SQLAlchemy	For mapping classes in the database.
PyOTP	A QRcode-based authentication for logging in registered users.
Flask-Captcha	Flask extension to validate human users and prevent bot attacks.
Visual studio code	For coding and hosting the code on GitHub.
Synk	For testing the app security.
Bleach	To reduce cross-site scripting attacks on the application.

Table 2. Logbook application library and tools.

System functionalities

The application aims to record ESA and associated researchers' sensitive and confidential astronaut logs. Therefore, the application must be secured against breaches yet be user-friendly, easy to use, and lightweight to reduce demands on systems and personnel. Table 3 illustrates four main functions.

Function	**Reason**
User registration	The application has user registration and login capabilities.
User roles	Astronauts can add or delete logs (Granted Privilege), administrators have Domain Admin Privileges, and researchers can view logs (Least Privilege). See Appendix. Use case.
CRUD functionality:	The system has three operations which are a server (Flask), user interface (browser, Flask), and database (SQlite3). The application performs CRUD via creating, reading, updating, and deleting users and logs.
Password protection	There are ten conditions in the sign-up function to improve password security (Owasp, 2017). See Appendix.

Table 3. System functions.

UML diagrams

The login application can be visualised to portray the system structure and behaviours (GeeksforGeeks, 2017). This document uses four diagram types (See Appendix B - H)

- Network: shows an overview of the system in a real-world setting.
- Sequence: highlights the interaction between actors and objects in sequential order.
- Use case: The system's functionality with actors (users) in a scenario.
- Class Diagram: depicts the login system.

Word count: 636

Appendices

Appendix A

Table 4 shows the functions of the password conditions in the Python programme.

Condition	**Reason**
2nd password	Improves memorisation of password
Email length <4	Reduces fake emails
First name length <3	Reduces fake or abbreviated names
Email as username	One ESA email for one user
One number in password	Increase security via complexity
One uppercase letter in password	Increase security via complexity
One lowercase letter in password	Increase security via complexity
One symbol in password	Increase security via complexity
Password length <8	Increase security via length

Table 4. Password protection conditions.

Appendix B

UML diagrams

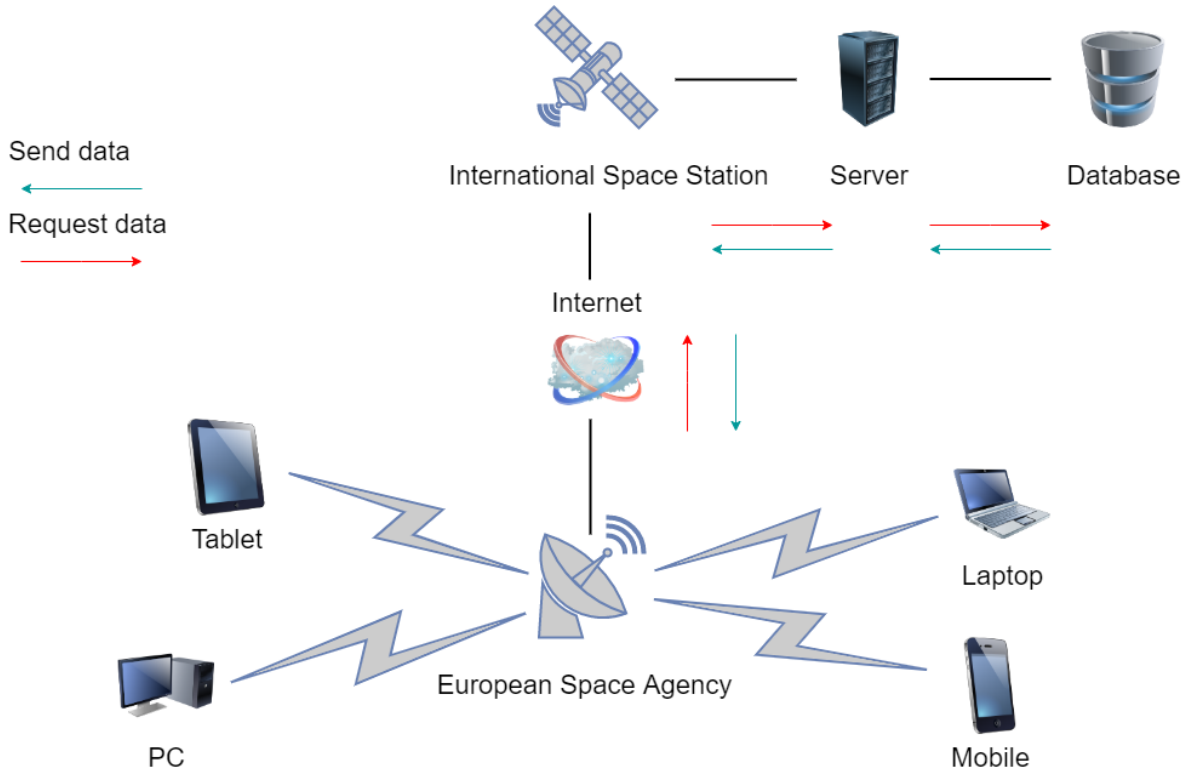


Figure 5. Logbook Network diagram. Adapted from diagrams.net (2022)

Appendix C

Sequence diagrams

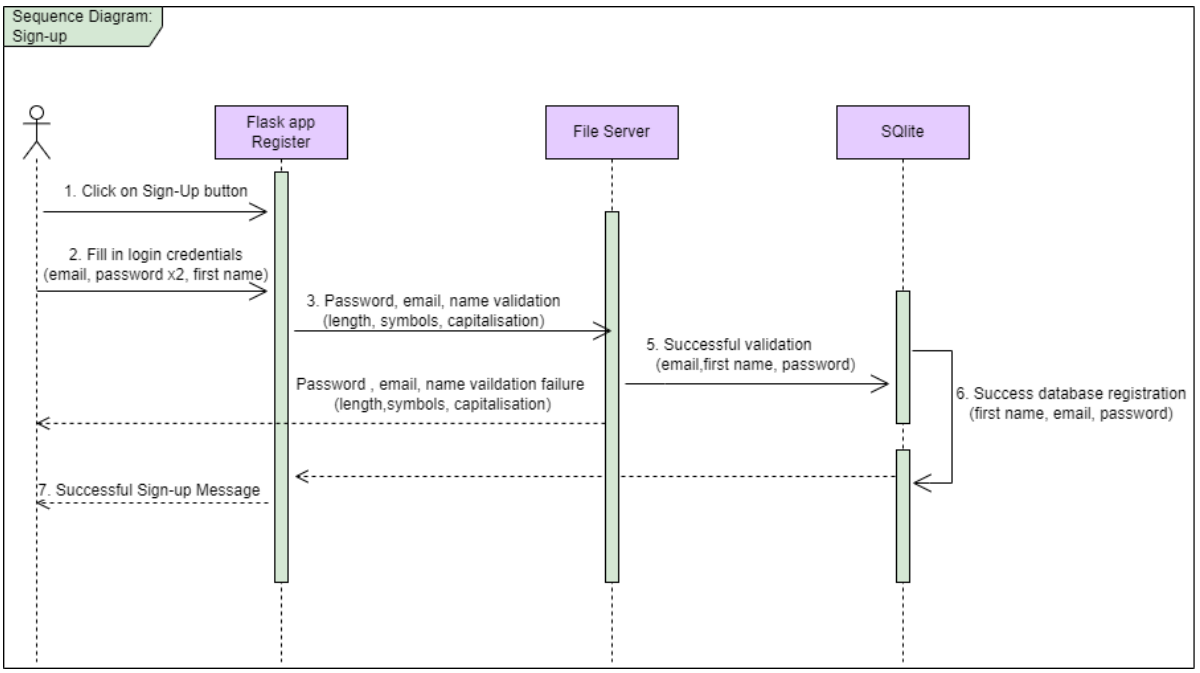


Figure 6. Sign-up Sequence diagram.

Appendix D

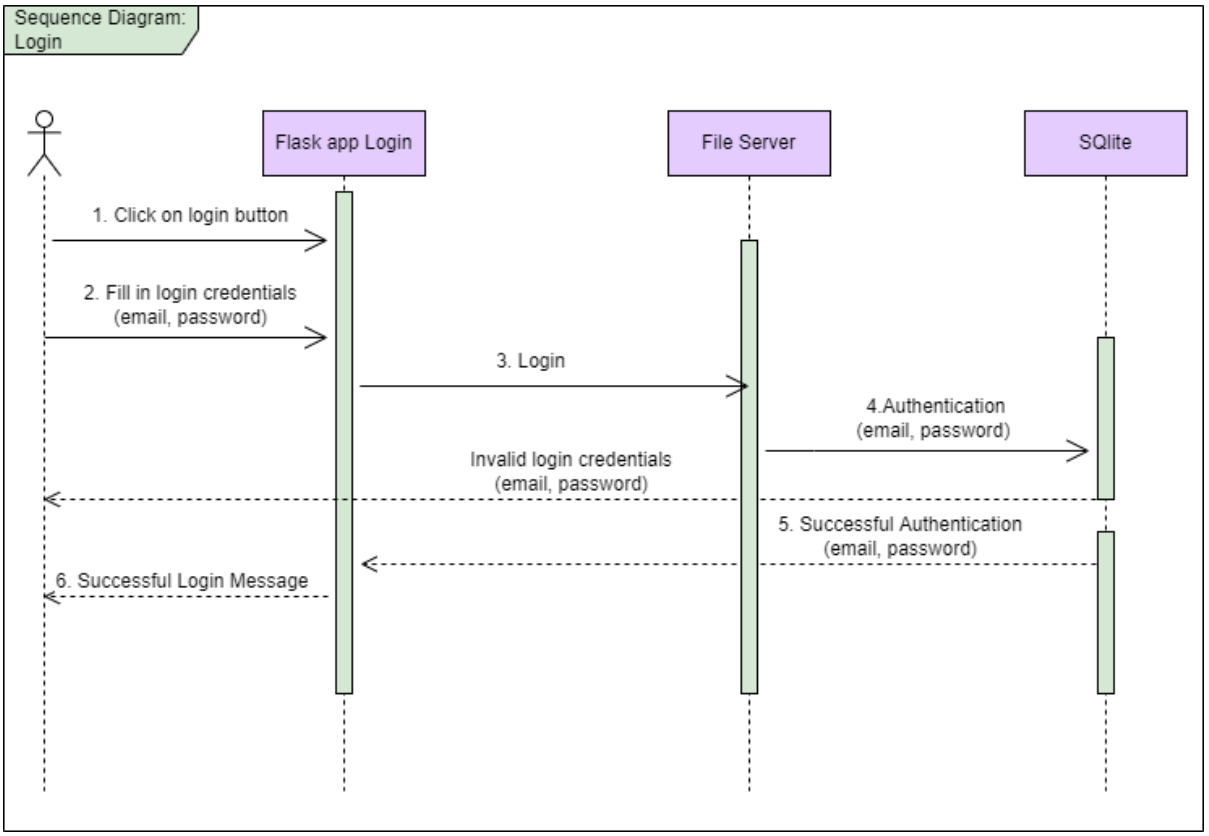


Figure 7. Login Sequence diagram. Adapted from Stack Overflow, (2016).

Appendix E

Application Flowchart

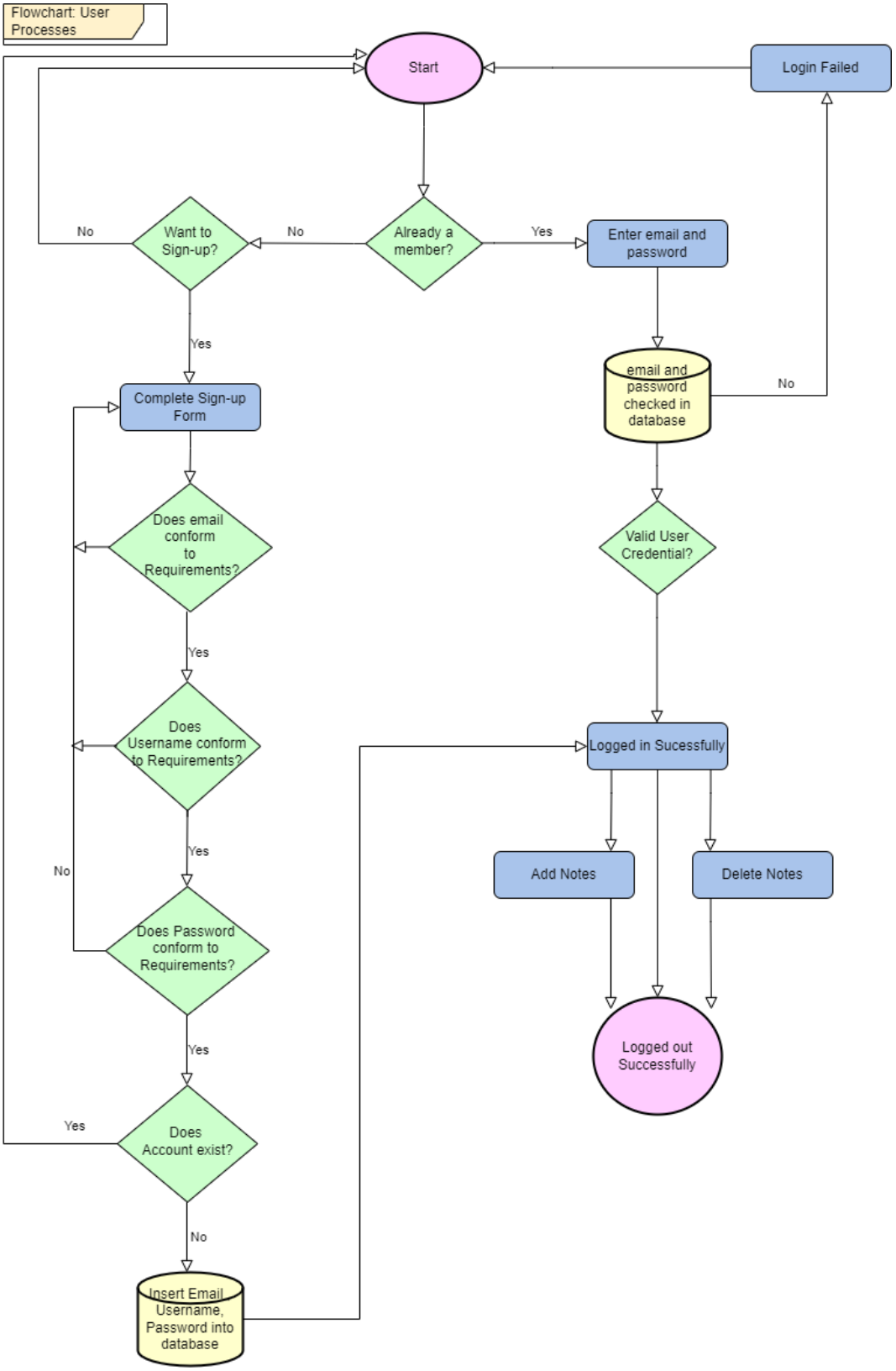


Figure 8. User Process Flowchart.

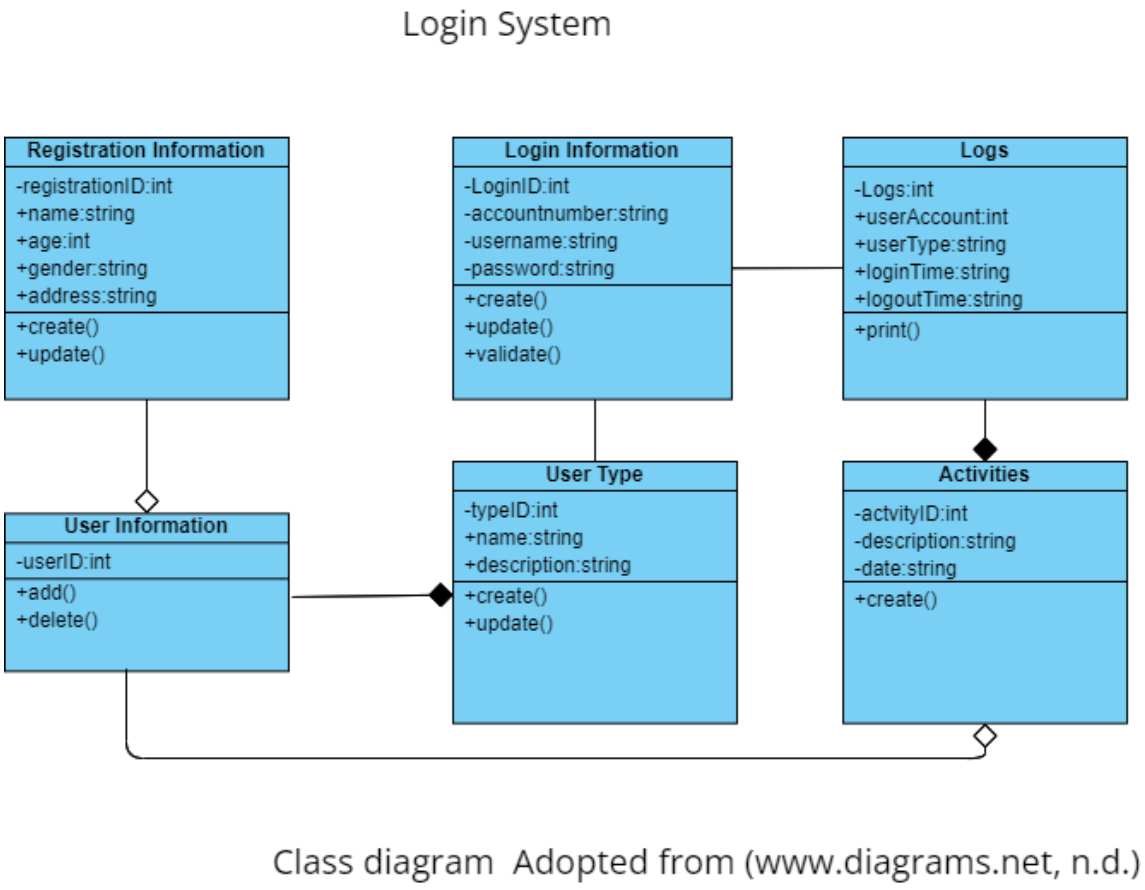
Appendix F

Use Case diagram

Figure 9. Use case diagram of the logbook system.

Appendix G

Class diagram



Class diagram Adopted from (www.diagrams.net, n.d.)

Figure 10.

Appendix H

Risk Assessment

Poor risk management has been counted as one of the major factors leading to software project failure, so the need to have a good risk assessment plan is crucial to the project's success (Ratsiepe & Yazdanifard, 2011).

Risk Assessment Approach

The qualitative risk assessment approach was used with the results shown using a likelihood/impact ranking matrix showing the risk factors which will be focused on before the commencement and during the project.

The impact, which is assessed in terms of low, medium, and high, is an estimate of the harm that a risk factor could cause, while the likelihood is assessed in terms of likely, unlikely, and very unlikely and is used to measure how probable the risk factor will occur.

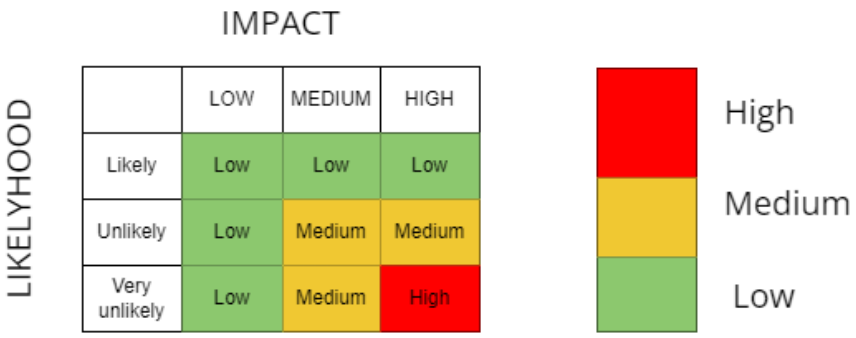


Figure 11.

The risks have been prioritised according to the top ten list of software risks (Boehm,1991) and (Addison, 2003), with the risk management technique included.

Risk Factors	**Risk Management Technique**	**Likelihood**	**Impact**	**Risk Level**
Unrealistic schedules and budgets	Developing and adhering to a software project plan to set realistic deadlines; multiple estimation techniques	Unlikely	High	Medium
Lack of user involvement	User testing and surveys	Unlikely	High	Medium
Frequent changes in requirements	Use of requirement scrubbing	Unlikely	High	Medium
Insufficient resources/less number of skilled employees	Contingency plan to cope with staffing problems.	Very Unlikely	High	High
Lack of senior management commitment and technical leadership	Implementing and using a communication plan	Unlikely	Medium	Medium
Developing the wrong software functions	Developing a prototype to be reviewed(tested) by the client	Very Unlikely	High	High

Incorrect system requirements	Feasibility Study of the system requirements	Unlikely	High	Medium
Lack of effective project management methodology	Use of good change management to control and manage the methodology.	Unlikely	High	Medium
Inadequate security features being put into the system	The use of standard framework to control the security implementation. (NIST 800-53,ISO 27002)	Unlikely	High	Medium
Failure to manage user expectation	User involvement , using quantitative risk analysis to evaluate the risk.	Unlikely	High	Medium

Table 5.

Appendix I

Cloud infrastructure (PaaS)

The main benefits of using the cloud:

1. Affordable access to a broader variety of resources

2. More freedom to test, with lesser risk

3. The application can be built and supported with less time spent provisioning it since no hardware or software is needed.

4. Proper scalability of resources

5. The platform-as-a-service model enables teams to work remotely in a shared software development environment with access to all the necessary tools.

References

Addison, Tom. (2003) E-commerce project development risks: evidence from a Delphi survey 1. International Journal of Information Management. 23. 25-40.

Boehm, B.W. (1991) Software risk management: principles and practices. *IEEE Software*, 8(1), pp.32–41. doi:10.1109/52.62930.

Crashtest-security. (2022) *Cryptographic Failures Vulnerability - Examples & Prevention*. Available from: https://crashtest-security.com/owasp-cryptographic-failures/#:~:text=A%20cryptographic%20failure%20is%20a [Accessed 18 Aug. 2022].

Diagrams.net. (2022) *Diagram Software and Flowchart Maker*. Available from: https://www.diagrams.net/ [Accessed 11 Aug. 2022].

Diagrams.net. (2022) *Example diagrams and templates*. Available from: https://www.diagrams.net/example-diagrams#uml-diagrams.

Esa.int. (2022) *Onboard Computers and Data Handling*. Available from: https://www.esa.int/Enabling_Support/Space_Engineering_Technology/Onboard_Computers_and_Data_Handling/Onboard_Computers_and_Data_Handling.

EU GDPR Compliance Criteria -Cybersecurity For Privacy (C4P) Overview Privacy Kick Off Cybersecurity Data Lifecycles START. (2021) *SecureControlsFramework*. Available from: https://www.securecontrolsframework.com/cybersecurity-materiality [Accessed 17 Aug. 2022].

GeeksforGeeks (2017) *Unified Modeling Language (UML) | An Introduction - GeeksforGeeks*. GeeksforGeeks. Available from: https://www.geeksforgeeks.org/unified-modeling-language-uml-introduction/ [Accessed 24 Aug. 2022].

Hewko, A. (2021) *STRIDE Threat Modeling | What Is It? | Explanation and Examples | Read*. Software Secured. Available from: https://www.softwaresecured.com/stride-threat-modeling/ [Accessed 24 Aug. 2022].

Kaplan-Moss, J. and Holovaty, A. (2008. *The Definitive Guide to Django: Web Development Done Right*. *Google Books*. Apress. Available from: https://www.google.co.uk/books/edition/The_Definitive_Guide_to_Django/u9UAPzCqLbQC?hl=en&gbpv=1&dq=Holovaty [Accessed 16 Aug. 2022].

Kaur, D. & Singh, H. (2012) Secure Spiral: A Secure Software Development Model Available from: https://www.researchgate.net/publication/272953501_Secure_Spiral_A_Secure_Software_Development_Model [Accessed 24 August 2022].

Learning Center. (2021) *What is OWASP | What are OWASP Top 10 Vulnerabilities | Imperva*. Available from: https://www.imperva.com/learn/application-security/owasp-top-10/ [Accessed 16 Aug. 2022].

Maier, P. (2017) Integrating Web Application Security into the Agile Software Development Process. Available from: https://diglib.tugraz.at/download.php?id=5a1deff264fec&location=browse [Accessed 26 Aug. 2022].

Nasa.gov. (2011) *NASA - Journaling the Journey into Space*. Available from: https://www.nasa.gov/mission_pages/station/research/news/Stuster_Journals.html [Accessed 14 Aug. 2022].

Navtor. (2022) *Digital Logbooks*. Available from: https://www.navtor.com/digital-logbooks [Accessed 14 Aug. 2022].

Outpost42.esa.int/. (2015) *Logbook | Outpost 42*. Available from: https://outpost42.esa.int/logbook/ [Accessed 14 Aug. 2022].

Owasp (2017) *Authentication · OWASP Cheat Sheet Series*. Available from: https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html [Accessed 24 Aug. 2022].

OWASP (2021) *OWASP Top 10:2021*. owasp.org. Available from: https://owasp.org/Top10/ [Accessed 14 Aug. 2022].

Ratsiepe, K.B. and Yazdanifard, R. (2011) *Poor Risk Management as One of the Major Reasons Causing Failure of Project Management*. IEEE Xplore. doi:10.1109/ICMSS.2011.5999104 [Accessed 25 Aug. 2022].

Reifer, D.J. (2006) *Software Management*. *Google Books*. John Wiley & Sons. Available from: https://www.google.co.uk/books/edition/Software_Management/9yrdvfZX3CoC?hl=en&gbpv=1&dq=For+the+software+application+development [Accessed 18 Aug. 2022].

Safran (2022) *An Introduction to Qualitative Risk Analysis | Safran*. www.safran.com. Available from: https://www.safran.com/content/introduction-qualitative-risk-analysis#qualitative-vs-quantitative-risk-analysis [Accessed 25 Aug. 2022].

Softscheck GmbH. (2022) *Agile Security*.Available from: https://www.softscheck.com/de/agile-security/ [Accessed 29 Aug. 2022]

Space Exploration Stack Exchange. (2014) *What kind of laptops do ISS astronauts use?* Available from: https://space.stackexchange.com/questions/1489/what-kind-of-laptops-do-iss-astronauts-use#:~:text=2%20Answers%201%20%20ThinkPad%20is%20the%20only [Accessed 16 Aug. 2022].

Stack Overflow. (2016) *uml - Sequence Diagram: Consuming Web Service from Android*. Available from: https://stackoverflow.com/questions/33722266/sequence-diagram-consuming-web-service-from-android [Accessed 15 Aug. 2022].